# NSight REST API Getting Started Guide

For release WiNG 5.9.3

# Table of Contents

# Preface

This section describes the text conventions used in this document, where you can find additional information, and how you can provide feedback to us.

## Text Conventions

Unless otherwise noted, information in this document applies to all supported environments for the products in question. Exceptions, like command keywords associated with a specific software version, are identified in the text.

When a feature, function, or operation pertains to a specific hardware product, the product name is used. When features, functions, and operations are the same across an entire product family, such as ExtremeSwitching switches or SLX routers, the product is referred to as *the switch* or *the router*.

**Table 1: Notes and warnings**

| Icon | Notice type | Alerts you to... |
|---|---|---|
| | Tip | Helpful tips and notices for using the product. |
| | Note | Useful information or instructions. |
| | Important | Important features or instructions. |

**Table 1: Notes and warnings (continued)**

| Icon | Notice type | Alerts you to... |
|------|-------------|------------------|
| ⚠️ | Caution | Risk of personal injury, system damage, or loss of data. |
| 🔺 | Warning | Risk of severe personal injury. |

**Table 2: Text**

| Convention | Description |
|------------|-------------|
| `screen displays` | This typeface indicates command syntax, or represents information as it appears on the screen. |
| The words *enter* and *type* | When you see the word *enter* in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says *type*. |
| **Key** names | Key names are written in boldface, for example **Ctrl** or **Esc**. If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press **Ctrl**+**Alt**+**Del** |
| *Words in italicized type* | Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles. |
| *NEW!* | New information. In a PDF, this is searchable text. |

**Table 3: Command syntax**

| Convention | Description |
|------------|-------------|
| **bold** text | Bold text indicates command names, keywords, and command options. |
| *italic* text | Italic text indicates variable content. |
| [  ] | Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets. |
| { **x** \| **y** \| **z** } | A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options. |
| **x** \| **y** | A vertical bar separates mutually exclusive elements. |
| < > | Nonprinting characters, such as passwords, are enclosed in angle brackets. |
| ... | Repeat the previous element, for example, `member[member...]`. |
| \ | In command examples, the backslash indicates a "soft" line break. When a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash. |

## Platform-Dependent Conventions

Unless otherwise noted, all information applies to all platforms supported by software, which are the following:

- ExtremeSwitching® switches
- SummitStack™

When a feature or feature implementation applies to specific platforms, the specific platform is noted in the heading for the section describing that implementation in the command documentation (see the Extreme Documentation page at www.extremenetworks.com/documentation/). In many cases, although the command is available on all platforms, each platform uses specific keywords. These keywords specific to each platform are shown in the Syntax Description and discussed in the Usage Guidelines sections.

## Providing Feedback

The Information Development team at Extreme Networks has made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you. We welcome all feedback, but we especially want to know about:

- Content errors, or confusing or conflicting information.
- Improvements that would help you find relevant information in the document.
- Broken links or usability issues.

If you would like to provide feedback, you can do so in three ways:

- In a web browser, select the feedback icon and complete the online feedback form.
- Access the feedback form at https://www.extremenetworks.com/documentation-feedback/.
- Email us at documentation@extremenetworks.com.

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

## Getting Help

If you require assistance, contact Extreme Networks using one of the following methods:

Extreme Portal

Search the GTAC (Global Technical Assistance Center) knowledge base; manage support cases and service contracts; download software; and obtain product licensing, training, and certifications.

The Hub

A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.

Call GTAC

For immediate support: (800) 998 2408 (toll-free in U.S. and Canada) or 1 (408) 579 2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number, or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any actions already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

## Subscribe to Service Notifications

You can subscribe to email notifications for product and software release announcements, Vulnerability Notices, and Service Notifications.

1. Go to www.extremenetworks.com/support/service-notification-form.
2. Complete the form (all fields are required).
3. Select the products for which you would like to receive notifications.

> **Note**
> You can modify your product selections or unsubscribe at any time.

4. Select **Submit**.

## Documentation and Training

Find Extreme Networks product information at the following locations:

Current Product Documentation
Release Notes
Hardware/software compatibility matrices for Campus and Edge products
Supported transceivers and cables for Data Center products
Other resources, like white papers, data sheets, and case studies

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For details, visit www.extremenetworks.com/education/.

# NSight REST API

NSight REST API documentation.

The NSight API enables developers to interact programmatically with the NSight database, configure NSight alarms, schedule reports, get troubleshooting information, and much more. It is based on RESTful principles and is organized around the main resources and features from the NSight platform's graphical user interface.

The NSight API uses a combination of configuration daemon and an HTTP/HTTPS front-end. API request and response bodies are formatted in JavaScript Object Notation (JSON). The front-end receives REST requests via standard HTTP/HTTPS methods and forwards the request to the NSight device after converting it into internal configuration daemon format. The response generated by the NSight configuration daemon is converted to JSON and sent back to the client.

To make API calls and build custom applications for users with an NSight account, you need to log in using credentials granting at least read permissions. Any administrator account can be used with the REST API, but only fully privileged accounts can be used to make configuration changes through the REST API.

> **Note**
> The NSight REST API is supported on all NX switches and NSight enabled devices.

This guide provides information about how to access the API, structure of the API request and response bodies, error codes, and usage examples.

> **Note**
> You cannot run the sample requests in this guide as-is. Replace call-specific parameters such as host IP address, user credentials, and session IDs with your own values.

Related Topics

## Accessing the NSight API

You can use any language or library that can submit REST API requests and process JSON to query the NSight API. Examples of languages and libraries that can build REST API clients include:

- For Java, the Jersey library provides the reference implementation of JAX-RS, a Java standard for RESTful web services. The implementation includes a client library that can run directly on the JVM.
- For Python, the Requests and JSON libraries facilitate REST API applications.
- For .Net, the core language provides facilities for submitting HTTP requests, and .Net libraries include a serializer for JSON.
- For the Linux shell, Wget and cURL can execute REST API calls. Linux shell utilities, like awk and grep, can parse and process JSON.

You can also use tools like Postman, an easy-to-use Chrome extension for making HTTP requests.

> **Note**
>
> The examples in this guide use cURL, a standard command line tool. All you need to do is replace call-specific parameters such as host IP address, user credentials, and session ID with your own values and you can test the calls from the command line.

# API Request Components

To construct a REST API request, combine the following components:

| Component | Description |
|---|---|
| The HTTP method | • GET: Retrieve data from the server<br>• DELETE: Delete a resource from the server<br>• POST: Create a new resource or update an existing resource on the server<br>• PATCH: Partially update a resource on the server<br><br>**Note:** The NSight API mainly uses the GET and POST methods to read data and update/edit existing data on the server respectively. |
| The base URL of the API | `http:// NSight_host_name_or_IP_address/ nsight-ui/api` |
| The URI to the resource | The resource to create, update, query, or delete. For example, `/v1/ap_stats` is the URI to retrieve access point statistics from the database. |
| Query parameters | These parameters, if any, appear after the question mark `(?)` in the API request. The question mark followed by the parameters and their values is referred to as the "query string." In the query string, each parameter is listed one right after the other with an ampersand `(&)` separating them. The order of the query string parameters does not matter. For example, `/v1/ alarmconfig? level=client&treeSelection=System` where `level` and `treeSelection` are query parameters used to filter response data based on their values. |
| HTTP request headers | The following HTTP headers are supported:<br>• Accept: Required for operations with a response body, syntax is `Accept: application/json`.<br>• Content-Type: Required for operations with a request body, syntax is `Content-Type: application/json`.<br>• Authorization: Required to get an access token or make API calls. |
| JSON request body | Required for most POST and PATCH requests. |

# API Response Codes

The NSight API returns standard HTTP status codes in addition to JSON-based error codes and messages in the response body.

**Table 4: HTTP Response Status Codes**

| Code | Description |
| --- | --- |
| 200 OK | The request was successful |
| 201 Created | The resource was created successfully |
| 204 No Content | Success with no response body |
| 400 Bad Request | The operation failed because the request is syntactically incorrect or violated schema |
| 401 Unauthorized | The authentication credentials are invalid or the user is not authorized to use the API |
| 404 Not Found | The server did not find the specified resource that matches the request URL |
| 405 Method Not Allowed | The API does not support the requested HTTP method |

## Error Codes and Messages

If an API request is successful, the response looks similar to the following example:

```
{
"success" : true,
"data" : <some data> - if the call returns data,
"return_code": 0
}
```

If an API request cannot be completed or results in an error, the response looks similar to the following example:

```
{
"success" : false,
"errors": <error description>,
"return_code": 1
}
```

# Authentication and Authorization

You must start a valid REST session by sending a basic authentication request to the NSight API server before you can start making API calls. The request should include a valid administrator user name and

password. The NSight server can use authentication mechanisms such as local database, RADIUS, etc, but the actual authentication mechanism(s) depends on the configuration of the WiNG device.

> **Note**
> NSight's REST API is protected by the same access restrictions which are provided via the WiNG command line or graphical user interface. For example, if a user role does not allow write access to a resource, then an attempt to configure/update this resource via REST will fail. For more information on NSight user roles, see the NSight User Guide on https://www.extremenetworks.com/support/documentation/.

## Sample Login Request

```
curl -L -H 'User-Agent: Mozilla' -H 'Cookie:troute=t1;' --cookie-jar ./nsight-cookie
'http://<NSight_Host_IP_Address>/nsight-ui/api/v1/login?
username=<username>&password=<password>'
```

| <NSight_Host_IP_Address> | The IP address of your NSight server |
|---|---|
| <username> | Your NSight account user name |
| <password> | Your NSight account password |

## Sample Login Response

```
{
    "success":true,
    "user":"admin",
    "role":"superuser",
    "session_id":"yskju15kx0nYVZ41t9ytcfcUMW9dBl3K",
    "fw_version":"5.9.3.0-6907683X",
    "serial_no":"7C47B72621E71E6E"
}
```

> **Note**
> The cookie is stored in the nsight-cookie file after you successful log in and needs to be used in subsequent API requests.

## Forwarding `session_id` in subsequent API calls

```
curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/ap_stats
```

You can send a logout request to the NSight API server to close a session.

> **Note**
> An idle REST session is terminated automatically by the WiNG device after the duration exceeds the `idle-session-timeout` value in the management policy. The default interval is 30 minutes.

## Sample Logout Request

```
curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/logout
```

## Sample Logout Response

```
{
    "data": {
        "status": "logged out"
    }
"success": true
}
```

> **Note**
> After you log out or if the session expires, you need to log in and start a new session to continue making API calls. You will see an error message if your session is no longer active.

## Sample Error Response if Session is not Active

```
{
    "success": false,
    "data":{
        "error': "Logged out"
        }
}
```

# Making Your First API Call

**Before You Begin**

**About This Task**

This sample API call demonstrates how to access the Info APIs and retrieve WLAN information.

To make REST API calls:

**Procedure**

1. Download cURL for your environment.

   > **Tip**
   > If you use Windows, use a Bash shell to make cURL calls.

2. Log in to the REST API server using administrator credentials.

   > **Note**
   > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

3. Use the GET method to access the `wlan_info` endpoint and fetch the IDs and corresponding WLAN names.

   **Sample Request**

   ```
   curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/wlan_info
   ```

Sample Response (200 OK)

```
{
  "data": [
  {
    "_id": 1039860226,
    "name": "nav-wpa",
    "ssid": "nav-wpa"
  },
  {
    "_id": 1249164677,
    "name": "nav-open",
    "ssid": "nav-open"
  },
  {
    "_id": 522874370,
    "name": "nav-eap",
    "ssid": "nav-eap"
  },
  {
    "_id": 1544565661,
    "name": "nav-5ghz",
    "ssid": "nav-5ghz"
   },
  {
    "_id": 753558181,
    "name": "rfs-wpa",
    "ssid": "rfs-wpa"
  },
  {
    "_id": 2108519761,
    "name": "4k-trend",
    "ssid": "4k-trend"
  },
  {
    "_id": 946589352,
    "name": "rfs-1",
    "ssid": "rfs-1"
  }],
"success": true
}
```

**What to Do Next**

For other common NSight API use cases, see API Usage Examples on page 18.

# List of API Endpoints

The NSight API provides the following endpoints to fetch data and configure various resources.

**Table 5: Info APIs**

| Method + URI | Description |
|---|---|
| **GET** `/v1/wlan_info` | This endpoint allows you to retrieve all WLAN IDs and corresponding WLAN names. |
| **GET** `/v1/avc_info` | This endpoint allows you to retrieve all application IDs and corresponding application names and categories. |
| **GET** `/v1/ap_info` | This endpoint allows you to retrieve all access point information either by MAC address or hostname. |

**Table 6: Statistics APIs**

| Method + URI | Description |
|---|---|
| **GET** `/v1/ap_stats` | This endpoint allows you to retrieve access point statistics. |
| **GET** `/v1/mu_wireless_stats` | This endpoint allows you to retrieve mobile unit wireless statistics. |
| **GET** `/v1/avc_stats` | This endpoint allows you to retrieve application visibility statistics. |
| **GET** `/v1/cpu_info` | This endpoint allows you to retrieve CPU statistics. |

**Table 7: Report and Alarms APIs**

| Method + URI | Description |
|---|---|
| **GET** `/v1/reports_config/templates` | This endpoint allows you to retrieve existing report configuration templates. |
| **GET** `/v1/alarmconfig/summary` | This endpoint allows you to retrieve a summary of configured alarms for site, device, or client. |
| **GET** `/v1/reports_config` | This endpoint allows you to retrieve information about reports that are scheduled and currently running. |

**Table 7: Report and Alarms APIs (continued)**

| Method + URI | Description |
|---|---|
| **POST** /v1/reports_config | This endpoint allows you to update a report configuration and schedule. |
| **GET** /v1/alarmconfig | This endpoint allows you to retrieve alarm configuration for site, device, or client. |

**Table 8: ASA (Advanced Spectrum Analysis) APIs**

| Method + URI | Description |
|---|---|
| **GET** /v1/tools/get-asa-preferences | This endpoint allows you to retrieve existing spectrum analysis preferences. |
| **POST** /v1/tools/set-asa-preferences | This endpoint allows you to update existing spectrum analysis preferences. |
| **GET** /v1/tools/get-asa-profiles | This endpoint allows you to retrieve existing spectrum analysis test suites/profiles. |
| **POST** /v1/tools/create-asa-profile | This endpoint allows you to create a spectrum analysis test suite/profile. |
| **POST** /v1/tools/update-asa-profile | This endpoint allows you to update a configured profile. |
| **POST** /v1/tools/delete-asa-profile | This endpoint allows you to delete a configured profile. |
| **GET** /v1/tools/get-asa-schedules | This endpoint allows you to retrieve existing spectrum analysis schedules. |
| **POST** /v1/tools/create-asa-schedule | This endpoint allows you to retrieve an ASA schedule. |
| **POST** /vq/tools/update-asa-schedule | This endpoint allows you to update a configured schedule. |
| **POST** /v1/tools/delete-asa-schedule | This endpoint allows you to delete a configured schedule. |
| **GET** /v1/tools/get-asa-reports-list | This endpoint allows you to retrieve a list of spectrum analysis reports. |
| **GET** /v1/tools/get-asa-report | This endpoint allows you to get a specific spectrum analysis report. |
| **POST**/v1/tools/delete-asa-reports | This endpoint allows you to delete ASA report(s). |

**Table 9: Access Point Test APIs**

| Method + URI | Description |
|---|---|
| **GET** /v1/tools/get-aptest-preferences | This endpoint allows you to retrieve the existing access point test preferences. |
| **POST** /v1/tools/set-aptest-preferences | This endpoint allows you to update the access point test preferences. |

**Table 9: Access Point Test APIs (continued)**

| Method + URI | Description |
|---|---|
| **GET** /v1/tools/get-aptest-profiles | This endpoint allows you to retrieve the existing access point test suites/profiles. |
| **POST** /v1/tools/create-aptest-profile | This endpoint allows you to create a new access point test suite/profile. |
| **POST** /v1/tools/update-aptest-profile | This endpoint allows you to update an access point test suite/profile. |
| **POST** /v1/tools/delete-aptest-profile | This endpoint allows you to delete an access point test suite/profile. |
| **GET** /v1/tools/get-aptest-schedules | This endpoint allows you to retrieve the existing access point test schedules. |
| **POST** /v1/tools/create-aptest-schedule | This endpoint allows you to create an access point test schedule. |
| **POST** /v1/tools/update-aptest-schedule | This endpoint allows you to update an access point test schedule. |
| **POST** /tools/delete-aptest-schedule | This endpoint allows you to delete an access point test schedule. |
| **GET** /v1/tools/get-aptest-result | This endpoint allows you to retrieve access point test reports. |
| **POST**/v1/tools/delete-aptest-requests | This endpoint allows you to delete one or more access point test reports. |

# API Usage Examples

This section provides information on how to accomplish a few common tasks using the NSight REST API.

Related Topics

# Get Access Point Information

**About This Task**

To retrieve the access point information:

**Procedure**

1. Log in to the REST API server using administrator credentials.

    > **Note**
    > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the GET method to access the `/ap_info` endpoint and fetch the access point information by either MAC address or hostname.

    Sample Request

    ```
    curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/ap_info?
    name=ap8533-5C21F1
    ```

    ```
    curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/ap_info?
    mac=74-67-F7-5C-21-F1
    ```

**Table 10: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|-----------|-------------------|-------------|-----------|
| mac | mac or name required | MAC address of device/wireless/ bluetooth interface | String |
| name | mac or name required | Hostname of the device | String |

Sample Response (200 OK)

```
{
"count": 1,
"data": [
{
    "ip4": "192.168.200.92",
    "ip6": [],
    "mac": "74-67-F7-5C-21-F1",
    "mac_type": "device",
    "name": "ap8533-5C21F1",
    "radio": [
      {
        "mac": "74-67-F7-71-62-F0",
        "mode": "2.4 GHz",
        "status": "on",
        "type": "wireless"
      },
      {
        "mac": "74-67-F7-6D-4F-10",
        "mode": "5 GHz",
        "status": "on",
        "type": "wireless"
      },
      {
```

```
          "mac": "74-67-F7-71-73-60",
          "mode": "sensor",
          "status": "off",
          "type": "wireless"
        }
        {
          "mac": "74-67-F7-5C-21-F5",
          "mode": "bt-sensor",
          "status": "off",
          "type": "bluetooth"
        }
      ],
      "rfd": "rfs",
      "state": "online",
      "type": "ap8533"
    }
  ],
"success": true
}
```

**Table 11: Response Parameters**

| Parameter | Description | Data Type |
|---|---|---|
| ip4 | IPv4 address | String |
| ip6 | IPv6 address | String |
| mac | MAC address of the device | String |
| mac_type | Type of the MAC provided in input. Device/Wireless/Bluetooth | String |
| name | Hostname of the device | String |
| radio.mode | Wireless radio mode or Bluetooth mode | String |
| radio.type | Wireless/Bluetooth | String |
| radio.status | Wireless/Bluetooth interface status | String |
| rfd | RF Domain | String |
| state | Device Status | String |
| type | Device Type | String |

# Get Access Point Statistics

**About This Task**

To retrieve the access point statistics:

**Procedure**

1. Log in to the REST API server using administrator credentials.

   > **Note**
   > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar`
   > option of the LOGIN API.

2. Use the GET method to access the `/ap_stats` endpoint and fetch the access point statistics.

Sample Request

```
curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/ap_stats
```

**Table 12: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|-----------|-------------------|-------------|-----------|
| count | Optional | The latest number of entries to be returned in the response. If you omit this parameter, the latest 100 access point entries are returned. | Integer |
| offset | Optional | The number of entries skipped in the sorted item list. The default value is 0. | Integer |

Sample Response (200 OK)

```
{
"count": 2,
"data": [
{
  "ap":[
  {
    "apid": "00236822A4D8",
    "cc_24": 0,
    "cc_5": 0,
    "cctotal": 0,
    "ch_util_24": 0,
    "ch_util_5": 0,
    "noise_24": 0,
    "noise_5": 0,
    "q_24": 0,
    "q_5": 0,
    "ri":
      [
        null,
        null
        ],
  "rssi_24": 0,
  "rssi_5": 0,
  "rssi_ri_24": 0,
  "rssi_ri_5": 0,
  "rtry_24": 0,
  "rtry_5": 0,
  "rtry_ri_24": 0,
  "rtry_ri_5": 0,
  "rx_24": 0,
  "rx_5": 0,
  "rxbps_24": 0,
  "rxbps_5": 0,
  "rxd_24": 0,
  "rxd_5": 0,
  "rxd_ri_24": 0,
  "rxd_ri_5": 0,
  "rxtotal": 0,
  "snr_24": 0,
  "snr_5": 0,
  "snr_ri_24": 0,
  "snr_ri_5": 0,
```

```
"t_24": 0,
"t_5": 0,
"tx_24": 0,
"tx_5": 0,
"txbps_24": 0,
"txbps_5": 0,
"txd_24": 0,
"txd_5": 0,
"txd_ri_24": 0,
"txd_ri_5": 0,
"txtotal": 0,
"usagetotal": 0,
"wired": [
  {
      "ifname": "up1",
      "rx": 0,
      "totrx": 0,
      "tottx": 0,
      "tx": 0
  },
  {
      "ifname": "ge5",
      "rx": 0,
      "totrx": 0,
      "tottx": 0,
      "tx": 0
  },
  {
      "ifname": "ge4",
      "rx": 0,
      "totrx": 116233337,
      "tottx": 287063276,
      "tx": 0
  },
  {
      "ifname": "ge3",
      "rx": 219614,
      "totrx": 1137602159,
      "tottx": 520035714,
      "tx": 1197285
  },
  {
      "ifname": "ge2",
      "rx":
      1714209,
      "totrx": 599461146,
      "tottx": 1485347811,
      "tx": 50294
  },
  {
      "ifname": "ge1",
      "rx": 9711,
      "totrx": 310967043,
      "tottx": 522568719,
      "tx": 17386
  }
  ]
  }
],
"cc_24": 0,
"cc_5": 0,
"cctotal": 0,
"ch_util_24": 0,
"ch_util_5": 0,
```

```
"i": 60,
"noise_24": 0,
"noise_5": 0,
"q_24": 0,
"q_5": 0,
"rfd": "rfs",
"rssi_24": 0,
"rssi_5": 0,
"rssi_ri_24": 0,
"rssi_ri_5": 0,
"rtry_24": 0,
"rtry_5": 0,
"rtry_ri_24": 0,
"rtry_ri_5": 0,
"rx_24": 0,
"rx_5": 0,
"rxbps_24": 0,
"rxbps_5": 0,
"rxd_24": 0,
"rxd_5": 0,
"rxd_ri_24": 0,
"rxd_ri_5": 0,
"rxtotal": 0,
"snr_24": 0,
"snr_5": 0,
"snr_ri_24": 0,
"snr_ri_5": 0,
"t_24": 0,
"t_5": 0,
"tx_24": 0,
"tx_5": 0,
"txbps_24": 0,
"txbps_5": 0,
"txd_24": 0,
"txd_5": 0,
"txd_ri_24": 0,
"txd_ri_5": 0,
"txtotal": 0,
"usagetotal": 0
},
{
"ap":[
{
  "apid": "7467F75C21F1",
  "cc_24": 0,
  "cc_5": 0,
  "cctotal": 0,
  "ch_util_24": 84,
  "ch_util_5": 21,
  "noise_24": -87,
  "noise_5": -92,
  "q_24": 0,
  "q_5": 0,
  "ri":
    [
      {
      "cc":0,
      "ch_util": 84,
      "noise": -87,
      "q":0,
      "rf-mode": 0,
      "riid": 1,
      "rssi": 0,
      "rtry": 0,
```

```
            "rx":0,
            "rxbps": 0,
            "rxd":0,
            "snr":0,
            "t":0,
            "tx":0,
            "txbps": 1,
            "txd":0,
            "wl":
            [
              {
                "cc":0,
                "rx":0,
                "tx":0,
                "wlid": "3DFB0202"
              },
              {
                "cc":0,
                "rx":0,
                "tx":0,
                "wlid": "4A74BD85"
              }
            ]
          },
          {
            "cc":0,
            "ch_util": 21,
            "noise": -92,
            "q": 0,
            "rf-mode": 1,
            "riid": 2,
            "rssi": 0,
            "rtry": 0,
            "rx": 0,
            "rxbps": 0,
            "rxd": 0,
            "snr": 0,
            "t": 0,
            "tx": 0,
            "txbps": 1,
            "txd": 0,
            "wl":[
              {
                "cc":0,
                "rx":0,
                "tx":0,
                "wlid": "3DFB0202"
              },
              {
                "cc":0,
                "rx":0,
                "tx":0,
                "wlid": "4A74BD85"
              }]
            }
            ],
          "rssi_24": 0,
          "rssi_5": 0,
          "rssi_ri_24": 0,
          "rssi_ri_5": 0,
          "rtry_24": 0,
          "rtry_5": 0,
          "rtry_ri_24": 0,
          "rtry_ri_5": 0,
```

```
          "rx_24": 0,
          "rx_5": 0,
          "rxbps_24": 0,
          "rxbps_5": 0,
          "rxd_24": 0,
          "rxd_5": 0,
          "rxd_ri_24": 0,
          "rxd_ri_5": 0,
          "rxtotal": 0,
          "snr_24": 0,
          "snr_5": 0,
          "snr_ri_24": 0,
          "snr_ri_5": 0,
          "t_24": 0,
          "t_5": 0,
          "tx_24": 0,
          "tx_5": 0,
          "txbps_24": 1,
          "txbps_5": 1,
          "txd_24": 0,
          "txd_5": 0,
          "txd_ri_24": 0,
          "txd_ri_5": 0,
          "txtotal": 0,
          "usagetotal": 0,
          "wired": [
          {
            "ifname": "ge2",
            "rx": 0,
            "totrx": 22230,
            "tottx": 25504,
            "tx": 0
          },
          {
            "ifname": "ge1",
            "rx":18396,
            "totrx": 533339920,
            "tottx": 316811241,
            "tx": 11162
          }]
        },
        {
          "apid": "B4C7996C87DB",
          "wired": [
          {
            "ifname": "xge4",
            "rx":0,
            "totrx": 0,
            "tottx": 0,
            "tx":0
          },
          {
            "ifname": "xge3",
            "rx": 0,
            "totrx": 0,
            "tottx": 0,
            "tx": 0
          },
          {
            "ifname": "xge2",
            "rx": 0,
            "totrx": 0,
            "tottx": 0,
            "tx": 0
```

```
              },
              {
                "ifname": "xge1",
                "rx": 0,
                "totrx": 0,
                "tottx": 0,
                "tx": 0
              },
              {
                "ifname": "ge2",
                "rx": 32423,
                "totrx": 8094700,
                "tottx": 23170262,
                "tx": 23894
              },
              {
                "ifname": "ge1",
                "rx": 0,
                "totrx": 0,
                "tottx": 0,
                "tx": 0
              }
          ]
        }],
      "cc_24": 0,
      "cc_5": 0,
      "cctotal": 0,
      "ch_util_24": 84,
      "ch_util_5": 21,
      "i": 60,
      "noise_24": -87,
      "noise_5": -92,
      "q_24": 0,
      "q_5": 0,
      "rfd": "test",
      "rssi_24": 0,
      "rssi_5": 0,
      "rssi_ri_24": 0,
      "rssi_ri_5": 0,
      "rtry_24": 0,
      "rtry_5": 0,
      "rtry_ri_24": 0,
      "rtry_ri_5": 0,
      "rx_24": 0,
      "rx_5": 0,
      "rxbps_24": 0,
      "rxbps_5": 0,
      "rxd_24": 0,
      "rxd_5": 0,
      "rxd_ri_24": 0,
      "rxd_ri_5": 0,
      "rxtotal": 0,
      "snr_24": 0,
      "snr_5": 0,
      "snr_ri_24": 0,
      "snr_ri_5": 0,
      "t_24": 0,
      "t_5": 0,
      "tx_24": 0,
      "tx_5": 0,
      "txbps_24": 1,
      "txbps_5": 1,
      "txd_24": 0,
      "txd_5": 0,
```

```
  "txd_ri_24": 0,
  "txd_ri_5": 0,
  "txtotal": 0,
  "usagetotal": 0
  }],
"success": true
}
```

**Table 13: Response Parameters**

| Parameter | Description | Data Type |
|-----------|-------------|-----------|
| count | Number of data items returned in output | Integer |
| total_count | Total number of data items present at the server | Integer |
| cc_24 | Client count on 2.4 GHz band | Integer |
| cc_5 | Client count on 5 GHz band | Integer |
| cctotal | CPU process utilization | Integer |
| ch_util_24 | Channel utilization on 2.4GHz band | Integer |
| ch_util_5 | Channel utilization on 5 GHz band | Integer |
| noise_24 | Noise floor on 2.4 GHz band in dB | Integer |
| noise_5 | Noise floor on 5 GHz band in dB | Integer |
| q_24 | Quality index on 2.4 GHz band | Integer |
| q_5 | Quality index on 5 GHz band | Integer |
| rssi_24 | RSSI in 2.4 GHz band in dB | Integer |
| rssi_5 | RSSI in 5 GHz band in dB | Integer |
| rssi_ri_24 | RSSI reference index in 2.4 GHz band in range 1-5 | Integer |
| rssi_ri_5 | RSSI reference index in 5 GHz band in range 1-5 | Integer |
| retry_24 | Retry count on 2.4 GHz band | Integer |
| retry_5 | Retry count on 5 GHz band | Integer |
| retry_ri_24 | Retries reference index in 2.4 GHz band in range 1-5 | Integer |
| retry_ri_5 | Retries reference index in 5 GHz band in range 1-5 | Integer |
| rx_24 | Receive octets on 2.4 GHz band | Integer |
| rx_5 | Receive octets on 5 GHz band | Integer |
| rxbps_24 | Receive rate on 2.4 GHz band in bps | Integer |
| rxbps_5 | Receive rate on 5 GHz band in bps | Integer |

**Table 13: Response Parameters (continued)**

| Parameter | Description | Data Type |
| --- | --- | --- |
| rxd_24 | Receive data rate on 2.4 GHz band in bps | Integer |
| rxd_5 | Receive data rate on 5 GHz band in bps | Integer |
| rxd_ri_24 | RX data rate reference index in 2.4 GHz band in range 1-5 | Integer |
| rxd_ri_5 | RX data rate reference index in 5 GHz band in range 1-5 | Integer |
| rxtotal | Total receive octets | |
| snr_24 | SNR on 2.4 GHz band in dB | Integer |
| snr_5 | SNR on 5 GHz band in dB | Integer |
| snr_ri_24 | SNR reference index in 2.4 GHz in range 1-5 | Integer |
| snr_ri_5 | SNR reference index in 5 GHz in range 1-5 | Integer |
| t_24 | Theoretical max speed reference index in 2.4 GHz in range 1-5 | Integer |
| t_5 | Theoretical max speed reference index in 5 GHz in range 1-5 | Integer |
| tx_24 | Transmit octets on 2.4 GHz band | Integer |
| tx_5 | Transmit octets on 5 GHz band | Integer |
| txbps_24 | Transmit rate on 2.4 GHz band in bps | Integer |
| txbps_5 | Transmit rate on 5 GHz band in bps | Integer |
| txd_24 | Transmit data rate on 2.4 GHz band in bps | Integer |
| txd_5 | Transmit data rate on 5 GHz band in bps | Integer |
| txd_ri_24 | Transmit data rate reference index in 2.4 GHz band in range 1-5 | Integer |
| txd_ri_5 | Transmit data rate reference index in 5 GHz band in range 1-5 | Integer |
| txtotal | Total tranmsit octets | Integer |
| usagetotal | Total usage octets | Integer |
| ri | Radio index (0 based) | Array |
| cc | Overall client count | Integer |
| ch_util | Overall channel utilization | Integer |
| noise | Overall noise floor in dB | Integer |

**Table 13: Response Parameters (continued)**

| Parameter | Description | Data Type |
|---|---|---|
| q | Overall quality index | Integer |
| rf_mode | Radio band in GHz | Integer |
| riid | Radio index | Integer |
| rssi | Overall RSSI in dB | Integer |
| rtry | Overall retry count | Integer |
| rx | Overall receive octets | Integer |
| rxbps | Overall receive rate in bps | Integer |
| rxd | Overall receive data rate in bps | Integer |
| snr | Overall SNR in dB | Integer |
| t | Overall theoretical max speed reference index in range 1-5 | Integer |
| tx | Overall transmit octets | Integer |
| txbps | Overall transmit rate in bps | Integer |
| txd | Overall transmit data rate in bps | Integer |
| wl | WLAN info | Array |
| cc | Client count | Integer |
| rx | Receive octets | Integer |
| tx | Transmit octets | Integer |
| wlid | WLAN index | String |
| wired | Wired Info | Array |
| ifname | Ethernet port name | String |
| rx | Receive octets | Integer |
| tx | Transmit octets | Integer |
| totrx | Total receive octets | Integer |
| totx | Total transmit octets | Integer |
| i | Entry index | Integer |

# Get Mobile Unit Wireless Statistics

**About This Task**

To retrieve the mobile unit wireless statistics:

**Procedure**

1. Log in to the REST API server using administrator credentials.

> **Note**
> You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the GET method to access the `/mu_wireless_stats` endpoint and fetch the mobile unit wireless statistics.

Sample Request

```
curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/
mu_wireless_stats
```

**Table 14: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|-----------|-------------------|-------------|-----------|
| count | Optional | The latest number of entries to be returned in the response. If you omit this parameter, the latest 100 access point entries are returned. | Integer |
| offset | Optional | The number of entries skipped in the sorted item list. The default value is 0. | Integer |

Sample Response (200 OK)

```
{
    "count": 1,
    "data": [
    {
        "mu":[
        {
            "apid": "7467F75C21F1",
            "apps":[
                {
                "id":"6311AE17",
                "req": 0,
                "rx": 0,
                "tx": 0
                }],
            "band": "2.4",
            "errors": 0,
            "keep_hist": 1,
            "muid": "B019C66BBECD",
            "noise": -87,
            "q": 2,
            "rssi": -85,
            "rssi_ri": 1,
            "rtry": 0,
            "rtry_ri": 5,
            "rx": 2392,
            "rxbps": 0,
            "rxd_ri": 1,
            "rxdrate": 13,
            "snr": 2,
            "snr_ri": 1,
            "tx": 0,
```

```
            "txbps": 0,
            "txd_ri": 1,
            "txdrate": 11,
            "wlid": "4A74BD85"
        }],
    "rfd": "test"
    }],
"success": true
}
}
```

**Table 15: Response Parameters**

| Parameter | Description | Data Type |
| --- | --- | --- |
| total_count | Total number of data items present at the server | Integer |
| count | Number of data items returned in output | Integer |
| muid | MU/Client MAC address | String |
| wlid | WLAN index | String |
| tx | Transmit octets | Integer |
| rx | Receive octets | Integer |
| txdrate | Transmit data rate in bps | Integer |
| rxdrate | Receive data rate in bps | Integer |
| txbps | Transmit rate in bps | Integer |
| rxbps | Receive rate in bps | Integer |
| keep_hist | Keep client history | Integer |
| rtry | Retry count | Integer |
| errors | Error count | Integer |
| rssi | RSSI value in dB | Integer |
| noise | Noise in dB | Integer |
| snr | SNR value in dB | Integer |
| q | Quality index | Integer |
| band | RF band | String |
| rssi_ri | RSSI reference index in range 1-5 | Integer |
| snr_ri | SNR reference index in range 1-5 | Integer |
| txd_ri | Transmit data reference index in range 1-5 | Integer |
| rxd_ri | Receive data reference index in range 1-5 | Integer |
| rtry_ri | Retries reference index in range 1-5 | Integer |
| apps | AVC applications | Array |

**Table 15: Response Parameters (continued)**

| Parameter | Description | Data Type |
|-----------|-------------|-----------|
| req | Required/Pinned application | Integer |
| id | Application index | String |

# Get Report Templates

### About This Task

To fetch the reports config templates:

### Procedure

1. Log in to the REST API server using administrator credentials.

   > **Note**
   >
   > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the GET method to access the `/reports_config/templates` endpoint and fetch the available reports template.

   **Sample Request**

   ```
   curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/
   reports_config/templates
   ```

   **Sample Response (200 OK)**

   ```
   {
   "data": {
       "report_template_list":[
       {
           "_id": "PCI_SYSTEM_TEMPLATE",
           "app": "nsight",
           "cb": "system",
           "rn": "PCI Compliance Report",
           "ro":[],
           "sh": true
       },
       {
           "_id": "5bbc9e55f4dab00d5a912228",
           "app": "nsight",
           "cb": "admin",
           "ri": null,
           "rn": "12D",
           "ro":[
            {
               "cf":[
               {
                   "name": "wlan",
                   "value": "all"
               },
               {
                   "appgrouptype": "appGroup",
                   "name": "appgroupapp",
                   "value": "trend"
               }],
   ```

```
            "oid":"CLIENTCOUNTTrend"
              }
            ],
            "sh": false
            }
          ]
      },
  "success": true
  }
```

**Table 16: Response Parameters**

| Parameter | Description | Data Type |
|---|---|---|
| app | The name of the application. It is always "nsight". | String |
| cb | The user name. | String |
| rn | The name of the report. | String |
| ro | The report object | Array |
| sh | Has the value `true` if the report is scheduled and the value `false` if it is not. | Boolean |
| cf | Filters | Object |
| oid | Object ID. | String |

# Update Report Configuration and Schedule

**About This Task**

To configure and schedule reports:

**Procedure**

1. Log in to the REST API server using administrator credentials.

   **Note**

   You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the POST method to access the `/reports_config` endpoint and configure and schedule reports.

   Sample Request

```
curl -X POST --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/
reports_config
-H "Content-Type: application/json"
-d {
    "title":"Daily Offline Devices",
    "type":"5bd2f389fbce720c74b5518e",
    "scopeType":"system",
    "list_id":"system",
    "siteGrpTree":"SiteGroup",
    "period":"2",
    "scheduled":false,
    "dom":null,
```

```
      "dow":null,
      "fmt":[
          true,
          false
      ],
      "dest":"1",
      "email":"support@extremenetworks.com",
      "_id":"",
      "ctx":"/System",
      "custrt":"",
      "cuend":"",
      "rcrtime":"18:30",
      "persist":"true",
      "runnow":"false",
      "pcidata":""
  }
  }
```

**Table 17: Request Body Parameters**

| Parameter | Description | Data Type |
|---|---|---|
| title | Report name | String |
| dest | The destination. 0=local, 1=email. | Integer |
| ctx | Context | String |
| email | Destination email | String |
| dom | Day of month | Integer |
| dow | Day of week | Integer |
| rcrtime | Recurrence time | String |
| fmt | Report format - 'pdf', 'xls' | Array |
| rn | Report template name | String |
| persist | Persistence flag | Boolean |

### Sample Response (200 OK)

The server returns the updated configuration.

```
{
"data":{},
"success": true
}
```

# Get Alarm Summary

### About This Task

To retrieve a summary of alarm statistics for a client, site, or device:

**Procedure**

1. Log in to the REST API server using administrator credentials.

   > **Note**
   > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the GET method to access the `alarmconfig/summary` endpoint and fetch the alarm statistics as a summary.

   Sample Request

   ```
   curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api
   /v1/alarmconfig/summary?treeSelection=%2FSystem&level=site&page=1&limit=30
   ```

**Table 18: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|---|---|---|---|
| treeSelection | Required | Used to filter the results in the response. Possible values are:<br>• /System: Returns results for all sites.<br>• /System/<site>: Returns results for a specific site. | String |
| level | Required | Site, device, or client | String |
| page | Optional | Number of pages | Integer |
| limit | Optional | Number of items per page | Integer |

   Sample Response (200 OK)

   ```
   {
   "success": true,
   "data":[
       {
           "_id":"1001",
           "active_alarms":1,
           "critical_alarms":0,
           "major_alarms":1,
           "minor_alarms":0
       },
       {
           "_id":"1002",
           "active_alarms":1,
           "critical_alarms":0,
           "major_alarms":1,
           "minor_alarms":0
        },
       {
           "_id":"1003",
           "active_alarms":1,
           "critical_alarms":0,
           "major_alarms":1,
           "minor_alarms":0
        },
       {
           "_id":"1004",
           "active_alarms":1,
   ```

```
         "critical_alarms":0,
         "major_alarms":1,
         "minor_alarms":0
     },
     {
         "_id":"1005",
         "active_alarms":1,
         "critical_alarms":0,
         "major_alarms":1,
         "minor_alarms":0
     },
     {
         "_id":"default",
         "active_alarms":1,
         "critical_alarms":0,
         "major_alarms":1,
         "minor_alarms":0
      },
     {
         "_id":"test",
         "active_alarms":1,
         "critical_alarms":0,
         "major_alarms":1,
         "minor_alarms":0
     }],
"total":7
}
```

## Get ASA Profiles

**About This Task**

To retrieve ASA profiles:

**Procedure**

1. Log in to the REST API server using administrator credentials.

   **Note**
   You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the GET method to access the `tools/get-asa-profiles` endpoint and fetch the ASA profiles.

   **Sample Request**

   ```
   curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api
   /v1/tools/get-asa-profiles?treeSelection=%2FSystem
   ```

**Table 19: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|---|---|---|---|
| treeSelection | Required | Used to filter the results in the response. Possible values are:<br>• /System: Returns results for all sites.<br>• /System/<site>: Returns results for a specific site. | String |

Sample Response (200 OK)

```
{
"data": [
{
    "2GHz": {
        "chR":[
            1,
            11
        ],
        "dcTh": -90,
        "th": -105
    },
    "5GHz": {
        "chR":[
            36,
            165
        ],
        "dcTh": -90,
        "th": -105
        },
    "_id": "5bc85cf966c0a60df0758e9d",
    "act": true,
    "chGN": "custom",
    "chts": [
        1,
        2,
        3,
        4,
        5
    ],
    "dt": 1000,
    "pName": "test",
    "rf": "both",
    "sch": "On Demand",
    "sd":{
        "dm": "",
        "dw": "",
        "edt": "",
        "rec": "One Time",
        "sdt": "",
        "t": ""
    },
    "st": 2,
    "user": "admin"
    }],
"success": true
}
```

**Table 20: Response Parameters**

| Parameter | Description | Data Type |
|-----------|-------------|-----------|
| chR | The channel range. | Integer Array |
| dcTh | The duty cycle threshold in dB. | Integer |
| th | The power threshold in dB. | Integer |
| act | Active flag. | Boolean |

**Table 20: Response Parameters (continued)**

| Parameter | Description | Data Type |
|-----------|-------------|-----------|
| chGN | The chart group name - Utilization/Physical Layer/ Interference/Spectrum detail/ custom | String |
| chts | List of charts. | Integer Array |
| dt | The dwell time. | Integer |
| pName | The profile name. | String |
| sch | The type of schedule. | String |
| sd | The schedule details. | Object |
| dm | Month | Integer |
| dw | Week | Integer |
| etd | End date | Timestamp |
| rec | Recurrence | String |
| t | Time | String |
| sdt | Start date | Timestamp |
| _id | The profile index. | String |

# Create ASA Schedule

**About This Task**

To create a spectrum analysis schedule:

**Procedure**

1. Log in to the REST API server using administrator credentials.

    > **Note**
    > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the POST method to access the `/create-asa-schedule` endpoint to configure the ASA schedule.

    Sample Request

```
curl -X POST --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/
create-asa-schedule?treeSelection=%2FSystem
    -H "Content-Type: application/json"
    -d {
        "2GHz": {
            "chR":[
                1,
                11
            ],
            "dcTh": -90,
            "th": -105
        },
```

```
        "5GHz": {
            "chR":[
                36,
                165
            ],
            "dcTh": -90,
            "th": -105
    },
    "act": true,
    "ap": "ap8533-5C21F1",
    "apid": [{
            "ap": "ap8533-5C21F1",
            "apid": "74-67-F7-5C-21-F1",
            "id": "extModel4069-3",
            "rfd": "test"
    }],
    "chGN": "custom",
    "chts": [
            1,
            2,
            3,
            4,
            5
    ],
    "dt": 1000,
    "en": "enable",
    "pName": "test",
    "rf": "both",
    "sch": "test1",
    "sd":{
            "dm": "",
            "dw": "",
            "edt": "2018-10-19",
            "rec": "Daily",
            "sdt": "2018-10-19",
            "t": "00:00"
    },
    "st": 2
}}
```

**Table 21: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|---|---|---|---|
| treeSelection | Required | Configures ASA schedule across all sites. Default value is /System. | String |

**Table 22: Request Body Parameters**

| Parameter | Description | Data Type |
|---|---|---|
| chR | The channel range. | Integer Array |
| dcTh | The duty cycle threshold in dB. | Integer |
| th | The power threshold in dB. | Integer |
| act | Active flag. | Boolean |

**Table 22: Request Body Parameters (continued)**

| Parameter | Description | Data Type |
|-----------|-------------|-----------|
| chGN | The chart group name - Utilization/Physical Layer/ Interference/Spectrum detail/ custom | String |
| chts | Charts | Integer Array |
| dt | The dwell time. | Integer |
| pName | The profile name. | String |
| rf | The RF mode. | String |
| sch | Schedule | String |
| dm | Month | Integer |
| dw | Week | Integer |
| edt | End date | String |
| rec | Recurrence | String |
| t | Time | String |
| sdt | Start date | Timestamp |

Sample Response (200 OK)

```
{
"success":true,
"return_code":0,
"err":"none"
}
```

# Get APTest Schedule

**About This Task**

To retrieve all configured access point test schedules:

**Procedure**

1.  using administrator credentials.

    > **Note**
    > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2.  Use the GET method to access the `tools/get-aptest-schedules` endpoint and fetch the access point test schedule.

    Sample Request

```
curl -X GET --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api
/v1/tools/get-aptest-schedules?treeSelection=%2FSystem
```

**Table 23: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|-----------|-------------------|-------------|-----------|
| treeSelection | Required | The type of access point entries to be returned in the response. Possible values are:<br>• /System: Returns access point schedules across all sites.<br>• /System/\<site>: Returns access point schedule for a specific site. | String |

Sample Response (200 OK)

```
{
"data":[{
        "_id": "5bc873430825060de16e1092",
        "active": true,
        "ap_count": 1,
        "apid": [
        {
            "apid": "74-67-F7-5C-21-F1",
            "apname": "ap8533-5C21F1",
            "next_t": 1540080000000,
            "rfd": "test",
            "rfmode": "2.4GHz-wlan",
            "ridx": 1,
            "state": 1,
            "status": 1,
            "widx": 1
        }],
        "apidlist":[
            "74-67-F7-5C-21-F1"
        ],
        "apname": "ap8533-5C21F1",
        "authentication": 1,
        "clientid": "",
        "dnsTest":{
            "dnsName": "",
            "nameLength": "",
            "options": "",
            "selected": false,
            "testCount": ""
        },
        "eap_params":{
            "fast_pac_certificate": "",
            "method": "",
            "outer_identity": "",
            "password": "",
            "server_certificate": "",
            "server_certificate_required": "",
            "user_certificate": "",
            "username": ""
        },
        "enable": "enable",
        "encryption": 5,
        "ipconfig":{
            "dnssettings": 0,
```

```
                            "domainName": "0.0.0.0",
                            "domainNameLength": 7,
                            "gateway": "0.0.0.0",
                            "ipaddr": "0.0.0.0",
                            "ipsettings": 0,
                            "netmask": "0.0.0.0",
                            "priDnsSrv": "0.0.0.0",
                            "reserved": "",
                            "secDnsSrv": "0.0.0.0"
                        },
                        "keyMgmtType": "",
                        "keySize": "",
                        "l2retry":{
                            "l2RetryCount": ""
                        },
                        "multicastEncType": 0,
                        "next_t": 1540080000000,
                        "ping": {
                            "addr": "192.168.200.1",
                            "count": 5,
                            "selected": true,
                            "size": 64,
                            "timeout": 1000
                        },
                        "portscanTest":{
                            "addr": "",
                            "addressLength": "",
                            "options": "",
                            "port": "",
                            "selected": false,
                            "testCount": ""
                        },
                        "profiles": "ping",
                        "psk_key": "",
                        "rfd":[
                            "test"
                        ],
                        "rfmode": "2.4GHz-wlan",
                        "ridx": "",
                        "schedule":{
                            "dayofmonth": "",
                            "dayofweek": "",
                            "enddate": "2018-10-21",
                            "recurrence": "Daily",
                            "startdate": "2018-10-21",
                            "time": "00:00"
                        },
                        "schedulename": "test1",
                        "sec_info": "",
                        "ssid": "nav-wpa",
                        "status": 1,
                        "thruputTest":{
                            "addr": "192.168.200.48",
                            "addressLength": 14,
                            "bwThreshold": 1,
                            "bwThresholdUnit": "kbps",
                            "filesize": 1,
                            "filesizeUnit": "MB",
                            "path": "test.txt",
                            "pathLength": 8,
                            "port": 21,
                            "pwd": "test1",
                            "pwdLength": 5,
                            "selected": true,
```

```
            "testDirection": 2,
            "timeOut": 30,
            "username": "user1",
            "usernameLength": 5,
            "xferType": 1
        },
        "tracerTest":{
            "addr": "192.168.200.1",
            "addressLength": 13,
            "options": 0,
            "selected": true,
            "tgtTestCount": 1
        },
        "treeSelection": "/System",
        "ts": "",
        "unicastEncType": 0,
        "user": "admin",
        "wep_params":{
            "keyIndex": "",
            "keyLength": "",
            "keyUnit": "",
            "keyValue": ""
        },
        "widx": "",
        "wlan": "nav-wpa",
        "wpa_protocol": null
    }],

"return_code":0,
"success":true,
}
```

## Delete APTest Profile

### About This Task
To delete an access point test profile:

### Procedure

1. Log in to the REST API server using administrator credentials.

   > **Note**
   > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the POST method to access the `tools/delete-aptest-profile` endpoint and delete an access point profile.

   Sample Request

```
curl -X POST --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/tools/
delete-aptest-profile?treeSelection=%2FSystem
    -H "Content-Type: application/json"
    -d {
        "profiles":"test 2",
```

```
        }
    }
```

**Table 24: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|---|---|---|---|
| treeSelection | Required | Deletes aptest profile based on filter value. Possible values are:<br>• /System: Deletes profile across all sites.<br>• /System/<site>: Deletes profile for a specific site. | String |

Sample Response (200 OK)

```
{
"success":true,
"return_code":0,
"err":"none"
}
```

# Update APTest Preferences

**About This Task**

To configure preferences for access point test reports:

**Procedure**

1. Log in to the REST API server using administrator credentials.

   > **Note**
   > You must forward the cookie stored in nsight-cookie file created using `--cookie-jar` option of the LOGIN API.

2. Use the POST method to access the `tools/set-aptest-preferences` endpoint and configure report preferences.

   Sample Request

```
curl -X POST --cookie ./nsight-cookie http://134.141.242.93/nsight-ui/api/v1/tools/
set-aptest-preferences?treeSelection=%2FSystem
    -H "Content-Type: application/json"
    -d {
        "pref":
            {
                "purge": 363,
                "id": extModel3026-1"
```

```
          }
     }
```

**Table 25: Query String Parameters**

| Parameter | Required/Optional | Description | Data Type |
|---|---|---|---|
| treeSelection | Required | The type of access point entries to be returned in the response. Possible values are:<br>• /System: Returns access point schedules across all sites.<br>• /System/<site>: Returns access point schedule for a specific site. | String |

**Table 26: Request Body Parameter**

| Parameter | Description | Data Type |
|---|---|---|
| purge | Duration in days after which reports will be purged. | Integer |

### Sample Response (200 OK)

```
{
"success":true,
"return_code":0,
"err":"none"
}
```